AGILE – НОВЫЙ ПОХОД К ЭФФЕКТИВНОЙ РАЗРАБОТКЕ ЗАКАЗНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Т.А. Костина
Государственный институт управления и социальных технологий БГУ
г. Минск, Беларусь

А.Д. Фан
Институт бизнеса и менеджмента технологий БГУ
г. Минск, Беларусь

Формально методологии разработки программного обеспечения можно разделить на два типа: классические (предсказуемые), т. е. все этапы детально проектируется заранее и не допускается отклонение от первоначального плана и адаптивные (гибкие), т. е. в процессе работы допустимо перепроектирование. Гибкие методологии или по-другому Agile-методологии появились в противовес классическим, детально описывающим процесс создания системы. Гибкая методология разработки – это серия подходов к разработке программного обеспечения, которые ориентированы на использование интерактивной разработки, динамическое формирование требований и обеспечение их реализации в результате постоянного взаимодействия внутри самоорганизующихся рабочих групп из специалистов различного профиля. Эта методология представляет собой попытку достичь необходимого компромисса между перегруженным процессом разработки и полным его отсутствием [1, с. 23–25].

Требования к программному обеспечению в процессе его создании в достаточной мере изменчивы. Таким образом, продукт и система его создания должны быть гибкими для своевременных изменений и учета новых требований. Методики Agile позволяют разрабатывать программное обеспечение в соответствии современным требованиям бизнеса.

Большинство Agile-команд собраны в одном офисе. Кроме того, команда включает и «заказчиков» или его полномочных представителей, определяющих требования к продукту. Эту роль может выполнять менеджер проекта, бизнес-аналитик или клиент. В офисе также располагаются тестировщики, дизайнеры интерфейса, технические писатели и менеджеры [1, с. 44–47].

Ключевой метрикой Agile-методов выступает рабочий продукт. Agile-методы уменьшают объем письменной документации по сравнению с классическими методами, поскольку отдают предпочтение непосредственному общению. Agile – это не единственный подход в разработке программного обеспечения, а комплекс процессов разработки. Этот метод не включает практик и определяет ценности и принципы для использования успешными командами.

В 2001 г. в феврале месяце на встрече независимых практиков методик программирования, именующих себя «Agile Alliance», были сформулированы основные пункты документа, содержащего описание ценностей и принципов гибкой разработки программного обеспечения – Agile-манифест. Манифест включает 4 основные идеи и 12 принципов [2, с. 53].

Основные идеи заключаются в следующих положениях [2, с. 57–58]:

– люди и взаимодействие важнее процессов и инструментов;

– работающий продукт важнее исчерпывающей документации;

– долгосрочное сотрудничество с заказчиком важнее согласования условий контракта;

– готовность к изменениям важнее следования первоначальному плану.

К принципам, которые разъясняет Agile, относятся [3]:

– удовлетворение клиента за счет ранней и бесперебойной поставки ценного программного обеспечения;

– приветствие изменений требований в том числе в конце разработки, что может повысить конкурентоспособность конечного продукта);

– частая поставка рабочего программного обеспечения (каждый месяц / неделю или еще чаще);

– тесное и ежедневное общение заказчика с разработчиками на протяжении всего срока выполнения проекта;

– проектом занимаются мотивированные личности, которые обеспечены нужными условиями работы, поддержкой и доверием;

– рекомендуемый метод передачи информации – личный разговор (лицом к лицу);

– работающее программное обеспечение – лучший измеритель прогресса;

– спонсоры, разработчики и пользователи должны иметь возможность поддерживать постоянный темп на неопределенный срок;

– постоянное внимание систематического улучшения технического мастерства и удобному дизайну;

– простота;

– лучшие технические требования, дизайн и архитектура получаются у самоорганизованной команды;

– постоянная адаптация к изменяющимся обстоятельствам.

В качестве преимуществ этого подхода можно выделить [3]:

– за короткий промежуток времени менеджер проекта и все члены команды могут оценить статус проекта;

– все проблемы решаются быстро, оперативно и доносятся до всех участников проекта, среди которых потенциально есть компетентные в данном вопросе специалисты;

– сотрудники учатся слушать других, понимать их, а также четко выражать свои собственные мысли;

– участники проекта учатся ставить перед собой реальные задачи и отвечать за статус их своевременного и качественного выполнения в соответствии с требованиями заказчика и стандартами.

В результате заказчик в любой момент видит выполненный этап разработки и может своевременно корректировать исходные характеристики продукта. Статистика улучшения качественных показателей в результате применения Agile-практик [4]:

Источник	**Результаты применения Agile-практик**	**К-во респондентов**
2008 Version One (Version One, 2008)[2]	76% - более 75% Agile-проектов являются успешными 56% - прирост 25 и более процентов показателя продуктивности и показателя time-to-market 55% - снижение 25 и более процентов стоимости 30% - снижение 25 и более процентов количества дефектов	2319
2007 University of Maryland University College, UMUC (Rico, 2007) [3]	26% - прирост качественных показателей на уровне 50 и более процентов	250
2006 Version One (Version One, 2006)	86% - рост показателя time-to-market 87% - улучшение показателей продуктивности 92% - высокая готовность к изменению приоритетов	722
2006 AmbySoft (Ambler, 2006)[4]	44%- улучшение стоимостных показателей, показателей продуктивности, качества 38% - рост показателя удовлетворенности клиентов	4232

Отдельная итерация Agile-метода представляет собой миниатюрный программный проект и включает все задачи, необходимые для выдачи прироста по функциональности [4]: планирование; анализ требований; проектирование; кодирование; тестирование; документирование.

В качестве особенностей Agile выступают: разделение рисков; предсказуемость; регулярная обратная связь; легкая и быстрая реакция на изменения; самоорганизация.

Что касается разделения рисков, надо заметить, что в настоящее время самые распространенные виды взаимодействия с заказчиком – это контракты с фиксированной ценой или контракты «время и материалы». В первом случае все риски по проекту ложатся на подрядчиков со всеми вытекающими последствиями. Согласно второй схеме заказчик оплачивает фактическую работу.

Предсказуемость в гибкой методологии разработки предполагает отказ от долгосрочного планирования, а также отсутствие цифр по срокам и стоимости продукта целиком. В начале итерации команда оценивает задачу и заранее обязуется предоставить результат.

Одна из главных проблем обратной связи – отсутствие взаимопонимания между заказчиком и подрядчиком по причине того, что первый не всегда может четко обозначить и сформулировать задачу.

Потерять свою актуальность может даже точно документированный проект после шести месяцев разработки. Перестройка исходной конструкции может привести к длительной переработке продукта. Постоянная динамика бизнеса не предоставляет лишнего времени на детальное документирование по причине изменчивости запросов пользователей.

Гибкие методологии подразумевают, что уже после первой итерации функциональность будет не полностью завершенной, и заказчик сможет вносить комментарии и поправки практически с самого старта проекта. Спустя несколько итерации для получения обратной связи с пользователями уже можно будет запускать первую версию продукта.

Самоорганизация в гибкой методологии разработки позволяет уйти от излишней структуры менеджмента, потому что существует только представитель заказчика, представляющий интересы бизнеса. Кроме того, нет необходимости в проверке каждого участника: команда сама распределит задачи и внутреннюю ответственность внутри, а также определенно будет гарантировать производительность труда и качество продукта. Наконец, самоорганизация здесь выступает в качестве мотивации команды.

Сравнение классических и гибких методологий приведено в таблице по таким позициям как процесс разработки, этапы и детализация планирования, степень вовлеченности заказчика, а также длительность сроков исполнения заказа и размер бюджета.

Таблица – Классические и гибкие методологии

Позиции	Классическая	Гибкая
Процесс разработки	Непрерывный, с заранее определенными этапами	Итерационный, количество итераций неизвестно
Планирование	Подробное один раз в начале проекта	Детализация по мере необходимости, многократное
Вовлечение заказчика	На этапе планирования	Постоянное
Сроки и бюджет	Заранее определены и прописаны в договоре	Неизвестны до самого конца проекта

К основным проблемам перехода на гибкие методологии можно отнести следующие шесть положений [5]:

1. Непонимание роли руководства при практическом внедрении методологии.

Переход на многие гибкие методологии подразумевает кардинальную смену задач и методов работы руководителей. Кратко описывая, этот переход можно определить как переход от управления к направлению, переход от приказов и указаний к рекомендациям.

Первая ошибка, которая допускается при таком переходе – это подсознательное стремление сохранить за собой власть, ведущее к тому же управлению. Вторая ошибка – это неправильное понимание этого перехода, что может привести к устранению менеджеров от руководящих функций, когда руководитель перестает давать указания, но так и не начинает давать рекомендации. Роль руководителя при этом сводится к чисто формальным, секретарским функциям. Причиной этого может быть и неготовность руководителей придавать аргументированность своему мнению (в старой модели им этого зачастую делать не приходилось) или боязнь невыполнения высказанных рекомендаций.

2. Построение «системы», не обладающей необходимой гибкостью и мобильностью.

Отсутствие опыта работы по новой методологии ведет к тому, что новый процесс внедряется в точности по инструкциям, по всем правилам, что ведет к негибкости и бюрократизации.

3. Начало внедрения не с «основ».

При переходе на новые методологии руководство преследует конкретные выгоды, которые должна обеспечить методология: высокая производительность, стабильное качество и т. п. При этом некоторые практики нового процесса более очевидно ведут к этим выгодам, а некоторые – не совсем ясно и очевидно. В связи с этим возникает соблазн и желание внедрить сначала более «выгодные» практики, а потом уже остальные. При этом не учитывается, что некоторые практики являются базовыми по отношению к другим, и внедрение вторых без первых с точки зрения эффективности невозможно.

4. Изменяются рабочие места, но не меняются привычки.

Провозглашение новых правил и следование этим правилам – это принципиально разные вещи. Недостаточное и неполное понимание новых идей всеми членами команды или слабое осознание выгод от перехода на новый процесс, слабая мотивация, отсутствие переходного периода с явно выраженными послаблениями в графике работ и количестве задач – это далеко не полный список ошибок, способных привести к тому, что новый процесс может соблюдаться лишь формально, т. е. «для видимости», а в действительности саботироваться и не выполняться членами команды.

5. Все измерять (собирать данные), но ни на что не реагировать.

Непрерывный мониторинг и анализ ситуации вместо постоянных улучшений. Большинство гибких методологий подразумевают сбор данных и обсуждение ошибок, допущенных на предыдущем этапе пе-

ред началом следующего. Распространенные ошибки в этой сфере – это сбор данных без последующего анализа или их неверная, слишком поверхностная интерпретация.

6. Обходиться без поддержки.

Отсутствие опыта работы команды по новой методологии и внедрение по букве инструкций содержит риск многих ошибок, неверных интерпретаций и недопонимания. Только участие в процессе перехода опытного инструктора, имеющего непосредственный опыт работы по новому процессу, может избавить команду от множества неверных решений и тупиков или в отдельных случаях даже от построения абсолютно неверной методологии.

Для того чтобы свести риски к минимуму при разработке заказного программного обеспечения необходимо постепенно подготавливать почву для внедрения Agile-методологий как на уровне разработки, так и на уровне руководства. Процесс внедрения гибких методологий подразумевает проведение командообразующих мероприятий и налаживание коммуникации между специалистами, посредством проведения практических тренингов.

В настоящее время существует не только гибкая методология Agile. Для сравнения приведем не менее актуальную модель разработки программного обеспечения – каскадную модель (Waterfall). При сравнении Waterfall и Agile необходимо осознание того, что Waterfall – это подход, хорошо описанный и детализированный. Agile – это, скорее, набор практик и принципов, которые поддерживают различные методологии гибкой разработки проектов.

Два подхода со своими плюсами и минусами, каждый из которых прекрасно подходит для применения в проектах с совершенно разными входными данными и требованиями [5].

Сравнительный анализ обеих моделей дает набор сильных сторон:

Waterfall	**Agile**
Легок для понимания и использования. Детально структурирован, что облегчает его применение к малоопытным командам. Задает стабильные требования к проекту/ продукту с самого старта. Проекты легко контролируются, отслеживаются ресурсы, риски, время. Качество имеет первоочередной приоритет по сравнению со стоимостью и временем	Итеративная разработка. Использование временные рамки (*time boxes*). Конечный пользователь вовлечен в процесс с самого начала. Быстрое получение первой / пробной версии продукта для тестирования. Легко воспринимаются корректировки и изменения в процессе разработки

Можно выделить набор слабых сторон обеих моделей:

Waterfall	**Agile**
Все требования должны быть определены и детально описаны до начала разработки. Дорого и медленно. Чувствителен к изменениям. Мало возможностей для конечного пользователя повлиять на цели проекта и требования к продукту. Зачастую проблемы выявляются на этапе тестирования. Много документации, много технической документации, которая не понятна конечному пользователю или заказчику	Может привести к низкому качеству продукта. Риск никогда не достигнуть закрытия / завершения проекта. Могут возникнуть проблемы с расширяемостью продукта

Сформулированы условия, которые определяют целесообразность использования моделей:

Waterfall	**Agile**
Требования к продукту предельно ясны и стабильны. Известны используемые технологии и инструменты. Проект большой, дорогой и сложный. Примеры: – внедрение новой версии известного продукта; – внедрение ERP систем	Конечный пользователь вовлечен в проект со старта. Четко определены бизнес-цели проекта / продукта. Небольшой или средний проект, относительно короткий по времени. Состав команды стабильный. Команда с высоким уровнем профессионализма. Технические требования приемлемые для технологий, которые собираются быть использованными для разработки. Система может быть модульной

Не исключена ситуация, когда обоими методами можно реализовать поставку одного и того же продукта, но на старте каждого из проектов входные данные по таким ключевым параметрам, как стоимость, время, квалификация команды, требования к качеству, конечный пользователь и т. д., существенно влияют на выбор методологии.

Задача руководителя проекта – выбрать наиболее подходящий способ для достижения целей проекта. Waterfall, Agile и другие методологии помогут этого добиться, если понимать разницу между ними,

их принципы, слабые и сильные стороны. В некоторых случаях, это не выбор между методологиями, а правильная комбинация подходов для каждого из этапов проекта. И, как показывает практика, Waterfall и Agile могут быть оптимально применены в рамках одного проекта.

Литература

1. *Thomas, D.* Agile Web Development with Rails. 2007. – P. 17–64.
2. Cohn M. Succeeding with Agile: Software Development Using Scrum. 2008. – P. 53–81.
3. Электронный ресурс. – 2012. – Режим доступа : https://ru.wikipedia.org/wiki/Гибкая_методология_разработки. – Дата доступа : 04.03.2015.
4. Электронный ресурс. – 2008. – Режим доступа : http:// versionone.com. – Дата доступа : 05.03.2015.
5. *Бэк, К.* Agile – манифест разработки программного обеспечения / К. Бэк, М. Бидли, Д. Гриннинг // [Электронный ресурс]. – 2014. – Режим доступа : http://agilemanifesto.org. – Дата доступа : 05.03.2015.